Hacia una Plataforma basada en la Web para la Programación Tangible

Celeste Ramos Jorge Rodríguez Laura Cecchi

Grupo de Investigación en Lenguajes e Inteligencia Artificial
Departamento de Teoría de la Computación - Facultad de Informática
UNIVERSIDAD NACIONAL DEL COMAHUE

{celeste.ramos, j.rodrig, lcecchi}@fi.uncoma.edu.ar

Resumen

En las últimas décadas, la currícula en la enseñanza de Ciencias de la Computación ha incorporado la utilización de distintas herramientas, que permiten a los estudiantes comenzar a programar. La interacción tangible mediada por el soporte de la computación, posibilita el alcance a estudiantes no alfabetizados, del aprendizaje de la resolución de problemas y de la programación.

La Programación Tangible permite la utilización de piezas físicas como componentes de programación. Cada pieza tiene características y funcionalidades específicas que al secuenciarlas es posible generar un programa.

Sin embargo, desarrollar Plataformas de Programación Tangible presenta actualmente, dificultades respecto de lo conceptual, metodológico y técnico.

Así, como primer contribución de este trabajo, se propone un modelo conceptual para Plataformas de Programación Tangibles, que especifica los componentes necesarios al diseñar e implementar dichas herramientas.

Alineados con el modelo propuesto, se diseñó e implementó un prototipo de una Plataforma Web, basada en un lenguaje de Programación Tangible nuevo.

Nuestro enfoque tiende al desarrollo de una herramienta, que permita a estudiantes no alfabetizados, sin experiencia previa en el uso de la computación, programar en dicho lenguaje. La herramienta tiene entre otras características ser de bajo costo, no requerir instalación, ser de licencia Open Source y utilizar objetos tangibles portables, flexibles y resistentes a grupos de niños.

Palabras Clave: Programación tangible - Enseñanza de la Programación - Educación en Ciencias de la Computación - Plataforma de Programación Tangible - Entorno Web para la Programación Tangible.

1. Introducción

En las últimas décadas, la enseñanza de las Ciencias de la Computación en el ámbito de la educación ha ido evolucionando. Progresivamente, la formación en ofimática está perdiendo relevancia dando lugar a la incorporación de tópicos que ayuden a desarrollar el Pensamiento Computacional [6, 7, 21].

El Pensamiento Computacional es una forma de trasponer prácticas y conceptos fundamentales de las Ciencias de la Computación para la resolución de problemas no necesariamente computacionales [28, 27]. Asimismo, el Pensamiento Computacional es considerado el núcleo de todas las disciplinas STEM modernas y es intrínseca a todas las otras disciplinas [9].

En este contexto, se están utilizando distintas herramientas que permiten a los estudiantes comenzar a programar alrededor de los 8 años.

Una de las formas más amigables para desarrollar el Pensamiento Computacional, en niños desde los 3 años de edad, es la Programación por medio de Interfaces de Usuario Tangibles(TUI) o simplemente Programación Tan-

gible(PT). Su fundamentación está en que los niños aprenden de una manera activa: aprenden a través de experiencias con otras personas, objetos y demás cosas que puedan percibir de su mundo.

Las TUI [4] son una forma de interacción hombre-máquina, novedosa y poco explorada, donde la interfaz se amplía al integrar la computación con objetos físicos y entornos que están vinculados a representaciones digitales. En definitiva, permiten que el usuario pueda manipular de forma física la información digital [15].

Desde hace más de una década, se han venido diseñando varios escenarios de aprendizaje basados en TUI, que han motivado el aprendizaje del pensamiento algorítmico y han hecho la programación más simple. Sin embargo, la mayoría de ellos no responde a un modelo, ni tampoco resuelve los desafíos que en la práctica se presentan [16, 19].

En este trabajo se presenta un modelo conceptual que permite orientar el desarrollo de Plataformas para la PT. Nuestro interés se centra en proveer un marco que contemple las características principales de dichas herramientas.

Alineados con esta idea, se introduce un nuevo lenguaje de PT. Su definición y diseño están enfocados a que sea comprensible para estudiantes no alfabetizados, de modo que puedan adquirir conceptos de programación y Pensamiento Computacional.

A diferencia de muchos de los lenguajes de PT existentes, se prioriza la utilización de objetos tangibles de bajo costo, accesibles, de uso cotidiano, resistentes y adecuados para ser manipulados por grupos de niños. Particularmente, se definió un sistema de códigos, que puede ser impreso en tarjetas, sobre cualquier material, como por ejemplo, cartulinas. La programación en este lenguaje, no requiere que los niños hayan tenido experiencias previas en la utilización de computadoras.

Finalmente, se presenta un prototipo en desarrollo de una plataforma Web con licencia Open Source, basada en el modelo propuesto, que permite a los estudiantes programar en el lenguaje definido.

Dicha plataforma es accesible desde cualquier navegador, por lo que sólo se necesita de un dispositivo con browser y conexión a internet, lo que permitirá su fácil acceso, sin necesidad de instalarla. Por otra parte, para su utilización no se requerirá de dispositivos de hardware específicos o exclusivos.

En la implementación presentada en este trabajo, se utiliza Realidad Aumentada, donde las tarjetas son los marcadores, que posibilitan visualizar los resultados del programa. Asimismo, se ofrece como alternativa a la tarjeta física en cartulina con el código, marcadores digitales dispuestos en cualquier dispositivo, como un teléfono móvil.

El trabajo presentado está estructurado como sigue. En la siguiente sección se describen los antecedentes y los trabajos relacionados, respecto de la Programación Tangible. En la sección 3, se expone el modelo conceptual propuesto, orientado a guiar el desarrollo de Plataformas para la PT. En la sección 4 se detallan aspectos relacionados al diseño e implementación y se presenta un prototipo de la Plataforma Web. Finalmente, se presentan las conclusiones y los trabajos futuros.

2. Programación Tangible: Marco Conceptual y Trabajos relacionados

El Pensamiento Computacional [28] es una de las habilidades fundamentales de este siglo, para el desarrollo de pensamiento analítico, la resolución de problemas, el diseño de sistemas y el entendimiento del comportamiento humano, haciendo transversales conceptos de la Ciencias de la Computación. Facultades como el análisis, la descomposición y el modelado de los aspectos relevantes de un problema complejo, son algunas de las capacidades que pueden adquirirse con este enfoque[12].

La Programación por medio de Interfaces de Usuario Tangibles o simplemente Programación Tangible, es una de las formas de desarrollo del Pensamiento Computacional en niños desde su primer infancia[23]. Las TUI[4] son un

paradigma de computación en el que el mundo real se ve incrementado al integrar la computación con objetos físicos y entornos que están vinculados a representaciones digitales.

Con los Lenguajes de Programación Tangibles los niños desde edades tempranas, tienen la oportunidad de manipular directamente objetos, ensamblando, encadenando y/o conectándolos. Estos objetos tangibles, en realidad, representan instrucciones precisas y generan, al secuenciarlos, un programa. Así, los niños interactúan con objetos físicos y transforman la lógica del mundo real en la lógica del programa. De este modo, se logra hacer que las manipulaciones simbólicas y abstractas involucradas en los procedimientos creativos, se vuelvan más concretas y manejables para los niños, a través de *tecnologías sin teclados*[5].

Actualmente, existen múltiples herramientas de PT[13] que permiten a los estudiantes aprender a programar desde los primeros años de edad. Tern y Quetzal[10], Tangicons[18], RoboBlocks[20], T-Maze[24], E-blocks[26], KIBO[22] y TanProRobot[25] son algunas de las herramientas que permiten a los niños programar robots físicos o personajes virtuales, al unir bloques de comando, ya sean de cartón, plástico, madera o electrónicos.

En los últimos años se han comenzado a combinar las TUI con la realidad aumentada. Ejemplos de estas herramientas son AR-Maze[11], que utiliza bloques como su antecesor T-Maze, pero los laberintos son marcadores que se recrean con realidad aumentada; Code Bits[8], que usa *bits de cartulina* para crear los programas y emplea la realidad aumentada para procesar el código; y AR-Scratch[14] que agrega realidad aumentada a las funcionalidades de la plataforma de programación Scratch, alcanzando una población de niños de entre 8 y 12 años. Los objetos tangibles son cartas y fichas redondas, marcadores en realidad aumentada, que permiten la aparición de sprites, haciendo que la creación de los programas se desarrolle en espacios que mezclan la realidad y la virtualidad.

Las herramientas mencionadas están adaptadas para la primer infancia, excepto por AR-Scratch, sin embargo, la gran mayoría tienen exclusividad de diseño (software y/o hardware propietarios) y por lo tanto, no pueden ser combinadas. Por otro lado, muchas de ellas tienen un alto costo y utilizan objetos tangibles que no son fácilmente asequibles y además difíciles de portar, lo que las hace poco accesibles para la gran mayoría de los niños y para las escuelas a las que éstos concurren.

En este sentido, algunos autores [16, 19], plantean algunos desafíos al desarrollar estas plataformas, como combinar objetos físicos y virtuales, la selección de múltiples acciones, reducir costos y aumentar la portabilidad y la disponibilidad de elementos del sistema. En general, las TUI, que utilizan tecnologías *sin teclado*, son más simples de usar y de comprender, pero este tipo de herramientas son más dificultosas para diseñar y construir, que las interfaces tradicionales, como las interfaces de usuario gráficas[19].

Así, los ejes presentados anteriormente describen una síntesis de características que los usuarios esperan de estas herramientas, marcando las bases teórica y práctica para el modelo que se propone en este trabajo.

3. Modelo Conceptual propuesto

En esta sección se presenta un modelo conceptual, desarrollado en el marco del estudio expuesto en este trabajo, que describen los componentes que necesita una Plataforma para la Programación Tangible.

Específicamente, el modelo conceptual propuesto se describe como una interacción compleja entre tres componentes: Entorno de Programación Tangible, Máquina Virtual de Lenguaje Tangible y Mundo.

El Entorno de Programación Tangible, habitualmente se integra de un Lenguaje, una colección de piezas físicas que implementan los elementos sintácticos de ese lenguaje, más un entorno donde el proceso de codificación sucede, en algunos casos se trata de un tablero digital, una mesa con características de iluminación específicas o simplemente un espacio demarcado

en el piso del aula.

La Máquina Virtual del Lenguaje Tangible, es una combinación específica de hardware y software capaz de interpretar un programa tangible y ejecutar las instrucciones sobre el mundo. La Máquina Virtual es una de pieza fundamental del modelo conceptual, se ubica en un nivel superior al Entorno de Programación Tangible y el Mundo sobre el que se pretende ejecutar el programa y actúa como enlace que interactúa entre estos componentes.

La incorporación del concepto de máquina virtual permite independizar el Lenguaje Programación Tangible (LPT) del mundo, de esta forma un programa puede ejecutarse sobre cualquier mundo para el cual exista una implementación de enlace. Es decir, esta característica hace posible ejecutar el mismo programa sobre un escenario digital, de realidad aumentada o físico siempre que tengan implementado un enlace con la Máquina virtual. El mismo concepto se aplica al lenguaje, así cualquier lenguaje que tenga implementado un enlace con la Máquina virtual puede ser interpretado y ejecutado sobre un mundo.

Se trata de un rasgo distintivo de este modelo conceptual, que no está presente en las plataformas estudiadas. En general se observa una fuerte ligazón entre el lenguaje, el mundo y la arquitectura de hardware.

El Mundo, es el componente que permite visualizar el resultado de la ejecución. Habitualmente las implementaciones utilizan escenarios digitales o físicos y recientemente agregan recursos de realidad aumentada.

Una característica esperada y observada en las herramientas estudiadas, tiene que ver con la implementación del concepto de mapeo directo de las operaciones sobre el escenario. Cada operación de un programa representa una acción observable en el mundo.

El concepto de mapeo directo se elabora en el ámbito de la enseñanza de la programación y está ampliamente difundido en los entornos basados en bloques, como dispositivo que busca reducir la carga cognitiva, ofrecer retroalimentación visible y confirmar la efectividad de programas y aprendizajes construidos.

Entorno de Programación Tangible

Los LPT asumen qué piezas físicas pueden constituirse en elementos sintácticos de un lenguaje (variables, instrucciones, operadores, entre otros). La colección de piezas físicas proporciona al programador, posiblemente un sujeto no alfabetizado, la posibilidad de secuenciar u ordenar elementos físicos para construir un algoritmo [13, 17].

Es posible pensar en LPT simple compuesto por un conjunto reducido de primitivas, por ejemplo que no involucra el uso de variables o de algunas estructuras de control, como un subconjunto de un LPT más complejo. Esta característica se convierte en andamiaje del proceso de aprendizaje, en el sentido que es posible sumar complejidad al lenguaje a medida que los nuevos conceptos sobre programación tienen posibilidades de integrarse a las estructuras cognitivas del sujeto.

En este modelo, un LPT puede tener diferentes implementaciones utilizando materiales diversos para la construcción de las piezas, aunque se promueve el uso de recursos de bajo costo como trozos de papel, cartulina u otros de uso habitual en la escuela.

Los estudiantes construyen programas en el LPT colocando una secuencia de tarjetas que representan las instrucciones del lenguaje.

Máquina Virtual del Lenguaje Tangible

En el momento en que el programa físico está elaborado, se digitaliza usando una cámara fotográfica convencional. La imagen capturada se procesa para producir un programa digital que luego se ejecuta. Finalmente, el resultado de la ejecución se mapea directamente sobre el mundo representado en forma digital, física o con recursos de Realidad Aumentada.

La Máquina Virtual está divida en tres etapas:

1. Digitalización del Programa Tangible (de ahora en más pt): es el proceso que permite la lectura del pt por medio de una cámara web o de un dispositivo móvil.

2. Procesamiento del Programa: es el proceso por el cual la versión digital del pt, esto es, una imagen, es representada en un lenguaje susceptible a ser ejecutado sobre el mundo.

3. Ejecución y Visualización de los Resultados del pt: realiza la ejecución del programa mostrando el comportamiento del agente dentro del mundo.

Mundo

Mayoritariamente el mundo se compone de elementos estáticos, es decir que no tienen un comportamiento asociado a la ejecución del programa tangible, además de uno o varios agentes que responden a instrucciones del lenguaje tangible. Constituye el espacio donde los estudiantes pueden visualizar el resultado de la ejecución de su código.

El mismo mundo puede tener diferentes representaciones, digitales o físicos o integrar recursos de realidad aumentada. Como el mundo está desacoplado de la Máquina Virtual, es posible adecuar la representación a la disponibilidad de recursos y a las opciones metodológicas definidas por el docente.

En el contexto de la experiencia de construir programas utilizando piezas físicas como proceso formativo, el estudiante formula hipótesis, pone en juego su creatividad e ingenio y experimenta intentando resolver problemas de codificación. Así, es importante que sea posible observar el resultado de la ejecución de sus programas, como forma de confirmar aprendizajes e identificar debilidades. Es decir, se espera que el mundo implemente el concepto de mapeo directo de las operaciones sobre el escenario.

4. Plataforma Web propuesta

En esta sección se exponen aspectos relacionados al diseño e implementación de la Plataforma Web de Programación Tangible, basado en el modelo propuesto, que permite a estudiantes no alfabetizados participar de experiencias tempranas de programación.

4.1. Aspectos relacionados al diseño

En esta primer etapa se definió y diseñó un LPT simple, prestando especial atención a la utilización de recursos de bajo costo y de una sintaxis que resulte accesible y familiar, a estudiantes no alfabetizados.

Dicha sintaxis está compuesta de símbolos que son intuitivos para los alumnos y que no requieren saber leer ni escribir para poder ser interpretados. Cada símbolo es un gráfico de color negro ubicado en el centro de un cuadrado, con bordes del mismo color y con fondo blanco, como se muestra en la Figura 1. La simplicidad, en la representación de estos símbolos, permite que puedan ser impresos, dibujados o construidos sobre piezas físicas, pudiendo utilizar materiales como papel, cartulina, cartón, etc.

Inicialmente, se definieron como símbolos flechas, con cuatro direcciones distintas, donde su semántica indica las acciones posibles a ejecutar por el bot, sobre un mundo representado en forma de cuadrícula. En este sentido las cuatro acciones iniciales definidas para el bot son: avanzar, retroceder, girar a la izquierda y girar a la derecha.

Un pt desarrollado en este lenguaje se interpreta de izquierda a derecha con sus instrucciones, piezas físicas, ubicadas en forma secuencial. En la Figura 1 se puede observar un pt representado por diez piezas. Cuando ese pt sea ejecutado por la Máquina Virtual, los movimientos del bot serán mapeados en el mundo como: avanzar tres casilleros, girar a la derecha, avanzar dos casilleros, luego girar a la izquierda, retroceder un casillero, girar a izquierda y finalmente, avanzar otro casillero.

Una vez definido el LPT, se diseñó la arquitectura del entorno Web. Esta arquitectura fue basada en el Modelo Vista Controlador (MVC), que permitió clasificar sus componentes y definir sus relaciones, como se puede observar en la Figura 2. Los componentes de esta arquitectura son:

- *Modelo:* contiene una representación de los lenguajes utilizados.

Figura 1: Ejemplo de un Programa en LPT

  - Lenguaje de Programación Tangible (LPT): está compuesto de símbolos impresos en piezas físicas, cuya semántica se corresponden con las acciones para el bot a manipular.
  - Lenguaje Intermedio (LI): es un lenguaje que, como su nombre lo indica, hace de intermediario entre el LPT y el mundo. El objetivo de utilizar este tipo de lenguaje es desacoplar el LPT de la forma en que se representa el mundo (interfaz de usuario). De esta forma se permite la interacción del LPT con distintas interfaces de usuarios que sean capaces de interpretar este LI y, de igual modo, utilizar la representación del mundo con cualquier lenguaje de programación que pueda ser mapeado al LI utilizado.

- *Vista:* representa al modelo en un formato adecuado para poder interactuar por medio de la interfaz de usuario. Tiene dos funciones:
  - *Obtener imagen del pt:* Consiste en obtener una imagen del programa definido por el usuario y enviarla al controlador.
  - *Mostrar acciones del bot:* se encarga de mostrar la ejecución de las acciones del bot en el mundo.

- *Controlador:* Es el encargado de responder a eventos y enviar peticiones al *modelo*, cuando se hace alguna solicitud desde la *vista* sobre la información que éste contiene. Además, notifica a la *vista* si hay algún cambio en el *modelo*.

  Respecto al diseño de la herramienta, contiene tres funcionalidades, que se corresponden con las indicadas en el marco del modelo conceptual, en la Máquina Virtual:
  - *Digitalización del pt:* Recibir la imagen de la vista para analizar su contenido y, en base a éste, generar el pt.
  - *Traducción a Lenguaje Intermedio (LI):* Una vez generado el pt, lo traduce al LI.
  - *Ejecución del pt:* Finalmente, interpreta el LI para poder ejecutar sus instrucciones en la interfaz de usuario.

4.2. Aspectos relacionados a la implementación del prototipo

A partir del diseño descripto se abordó la implementación de la herramienta, la que fue dividida de acuerdo a las funcionalidades:

- *Digitalización del pt:* el objetivo es la implementación de una aplicación que permita leer las tarjetas que componen a un pt. Esto se logró utilizando una cámara convencional (como la que contienen celulares, tabletas, etc) y herramientas de Realidad Aumentada, como A-Frame y AR js [1, 2].

  A-Frame es un framework Web de código abierto para el desarrollo de ambientes en realidad virtual que utiliza Javascript y HTML. Si se lo usa en conjunto con AR js, puede utilizarse para el desarrollo de aplicaciones en realidad aumentada. AR js es una librería de JavaScript para Realidad Aumentada en la Web, que posibilita implementar el seguimiento de marcadores. Esto permitió poder utilizar cada tarjeta del

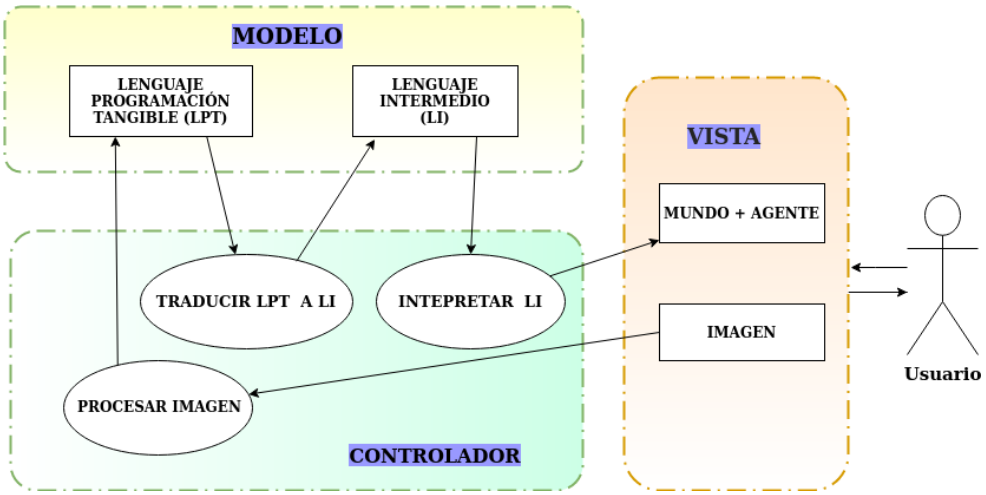

Figura 2: Diseño de la herramienta con el Modelo Vista Controlador (MVC)

pt como un marcador que pueda ser interpretado por estas herramientas.

Una vez que la tarjeta (o marcador) es detectada por la cámara (vista) se genera un evento que es capturado por el controlador, para luego realizar un mapeo entre la tarjeta leída y el movimiento del agente asociado.

Como resultado de este mapeo se observa sobre la tarjeta un objeto 3D, que representa el movimiento del agente para dicha tarjeta. En la Figura 3a, se indica que el agente debe avanzar un casillero hacia adelante, y en la Figura 3b el agente debe girar hacia la derecha (sentido horario).

- *Traducción a LI:* Una vez que todo el programa es leído, es decir, que todas las tarjetas del pt son capturadas y mapeadas a un movimiento del agente, el controlador traduce este resultado a un LI. Para este desarrollo se optó por representar el LI con el formato de texto JSON[3] (JavaScript Object Notation), ya que es una notación simple y muy utilizada para el intercambio de datos.

- *Ejecución del pt:* el objetivo es la ejecución del pt, donde se puede observar el comportamiento del agente dentro del mundo.

  Actualmente, el personaje realiza el movimiento correspondiente al marcador detectado, sin estar inserto en un mundo particular.

4.3. Interfaz con el Usuario en la detección del pt: Ejemplo de Uso

El proceso de digitalización del pt ocurre de forma paulatina, leyendo una a una las tarjetas del programa creado por el usuario.

En la Figura 4 se presenta una captura de pantalla, donde se muestra la interfaz de usuario, a medida que se detectan cada una de las tarjetas. En la parte superior, se observa lo que va capturando la cámara del dispositivo. Por otro lado, en la parte inferior izquierda de la pantalla se visualiza el pt con las tarjetas que ya han sido detectadas.

En este ejemplo, se puede ver una tarjeta (o marcador), que está cargada en un celular, indicando que el agente debe realizar un movimiento hacia atrás. Una vez que el marcador es detectado por la cámara, en la parte superior de la pantalla, comienza a reproducirse un personaje 3D, generado por las herramientas de realidad aumentada, realizando el movimiento asociado a la tarjeta. Mientras que en la parte inferior, se agrega la misma tarjeta al final de la secuencia del pt detectado.

Cuando se completa la lectura del pt, se presiona el botón verde “ejecutar” para que la in-

(a) Movimiento Avanzar. (b) Movimiento Giro a Derecha.

Figura 3: Lectura de dos tarjetas de un pt con Realidad Aumentada

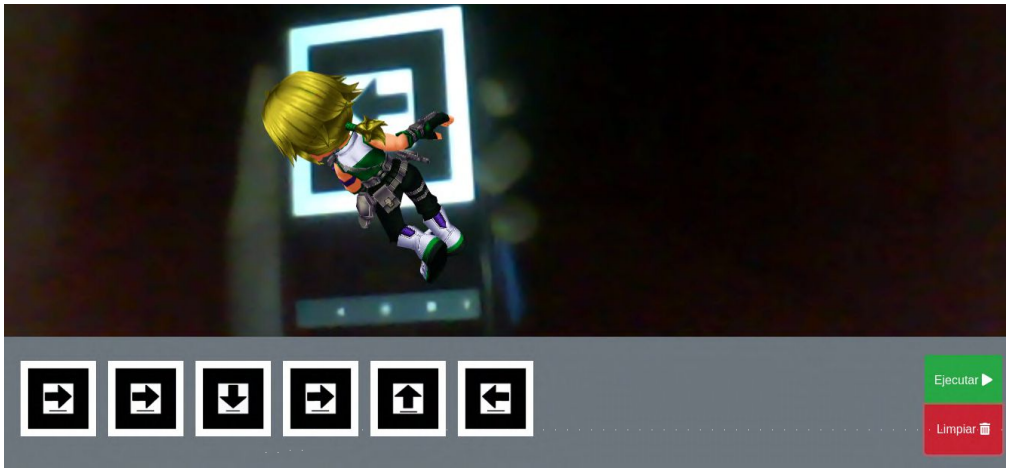

Figura 4: Ejemplo Interfaz Usuario a medida que se leen las tarjeta del pt.

formación del pt obtenido sea reproducida por el agente en el mundo.

Si, por el contrario, se quiere empezar nuevamente a construir otro pt, entonces se debe presionar el botón rojo “limpiar” para vaciar el contenido del pt cargado.

5. Conclusiones y Trabajo Futuro

Las TUI hacen a la programación más atractiva para niños de todas las edades, desde los 3 años, ayudando a un mejor abordaje y entendimiento de la resolución de problemas[13].

Las diferentes herramientas que se han desarrollado para la Programación Tangible presentan características diferentes, aunque en general, todas fomentan la facilidad y simplicidad en su uso.

A partir de este marco es que la primer contribución de este trabajo es un modelo conceptual de Plataformas de Programación Tangible que pretende ser un marco de referencia para diseñadores, y que aborda varias de las problemáticas planteadas al momento de construir estas herramientas. Bajo este modelo, se especifican y describen los componentes que caracterizan la construcción de una plataforma. Específicamente, el modelo conceptual propuesto se describe como una interacción compleja entre tres componentes: Entorno de Programación Tangible, Máquina Virtual de Lenguaje Tangible y Mundo. La división en estos tres aspectos es fundamental para generar independencia entre el LPT, el sistema que lo soporta y el mundo, donde las acciones se llevan a cabo.

Asimismo, presentó el diseño de una Plataforma Web para la PT, que está alineada con el modelo conceptual y se describieron los aspectos más relevantes de la implementación del prototipo.

En este sentido, se introdujo un nuevo LPT, basado en flechas que puede ser impreso sobre cartulinas o tener los símbolos en dispositivos móviles. A futuro, se pretende experimentar con los símbolos impresos con impresoras 3D.

Actualmente, se está diseñando el mundo donde el personaje se moverá. Asimismo, se están evaluando las tecnologías que se utilizarán sobre la interfaz de usuario, para mostrar el mundo y los movimientos que el agente deba realizar, según lo especificado en el LI generado.

Se espera que una plataforma desarrollada bajo este modelo contribuya a simplificar y enriquecer los procesos de enseñanza y de aprendizaje de la programación, alcanzando una población no alfabetizada. Se busca evitar que los estudiantes interactúen con la sintaxis de un lenguaje de programación textual. Por otra parte, al ser una herramienta Web se eliminan los aspectos relacionados a la instalación y configuración del sistema, sin demandar recursos costosos ni específicos.

Referencias

[1] A-FRAME Homepage `https://https://aframe.io/`. Accedido por última vez Marzo 2020.

[2] AR-JS Homepage `https://github.com/AR-js-org/AR.js`. Accedido por última vez Marzo 2020.

[3] JSON Homepage `https://www.json.org/json-es.html`. Accedido por última vez Marzo 2020.

[4] A. N. Antle and A. Wise. Getting down to details: Using learning theory to inform tangibles research and design for children. *Interacting with Computers*, 25(1):1–20, 2013.

[5] J. Cassell. Towards a model of technology and literacy development: Story listening systems. *Journal of Applied Developmental Psychology*, 25(1):75–105, 2004.

[6] C. P. de Educación de la Provincia de Neuquén. Diseño Curricular Jurisdiccional de los tres primeros años de la Escuela Secundaria Neuquina. Resolución N°1463/18, 2018.

[7] S. Furber. *Shut down or restart? The way forward for computing in UK schools*. The Royal Society Education Section, 2012.

[8] S. Goyal, R. S. Vijay, C. Monga, and P. Kalita. Code bits: an inexpensive tangible computational thinking toolkit for k-12 curriculum. In *Proceedings of the TEI'16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction*, pages 441–447, 2016.

[9] P. B. Henderson, T. J. Cortina, and J. M. Wing. Computational thinking. In *Proceedings of the 38th SIGCSE technical symposium on Computer science education*, pages 195–196, 2007.

[10] M. S. Horn, E. T. Solovey, R. J. Crouser, and R. J. Jacob. Comparing the use of tangible and graphical programming languages for informal science education. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 975–984, New York, NY, USA, 2009. ACM.

[11] Q. Jin, D. Wang, X. Deng, N. Zheng, and S. Chiu. Ar-maze: a tangible programming tool for children based on ar technology. In *Proceedings of the 17th ACM Conference on Interaction Design and Children*, pages 611–616, 2018.

[12] I. Lee, F. Martin, J. Denner, B. Coulter, W. Allan, J. Erickson, J. Malyn-Smith, and L. Werner. Computational thinking for youth in practice. *Acm Inroads*, 2(1):32–37, 2011.

[13] S. Papavlasopoulou, M. N. Giannakos, and L. Jaccheri. Reviewing the affordances of tangible programming languages: Implications for design and practice. In *2017 IEEE Global Engineering Education Conference (EDUCON)*, pages 1811–1816. IEEE, 2017.

[14] I. Radu and B. MacIntyre. Augmented-reality scratch: a tangible programming.[en]vironment for children. In *Proceedings of Conference on IDC*, 2009.

[15] J. M. Ruzafa. *Una arquitectura para aplicaciones educativas basadas en mundos virtuales e interfaces tangibles*. PhD thesis, Universidad Autónoma de Madrid, 2018.

[16] T. Sapounidis, I. Stamelos, and S. Demetriadis. Tangible user interfaces for programming and education: A new field for innovation and entrepreneurship. In *Advances in Digital Education and Lifelong Learning*, volume 2, pages 271–295. Emerald Group Publ. Ltd, 2016.

[17] T. Sapounidis, I. Stamelos, and S. Demetriadis. Tangible user interfaces for programming and education: A new field for innovation and entrepreneurship', innovation and entrepreneurship in education (advances in digital education and lifelong learning, volume 2), 2016.

[18] F. Scharf, T. Winkler, and M. Herczeg. Tangicons: algorithmic reasoning in a collaborative game for children in kindergarten and first class. In *Proceedings of the 7th IDC*, pages 242–249, 2008.

[19] O. Shaer and R. J. Jacob. A specification paradigm for the design and implementation of tangible user interfaces. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 16(4):1–39, 2009.

[20] A. Sipitakiat and N. Nusen. Robo-blocks: Designing debugging abilities in a tangible programming system for early primary school children. In *Proceedings of the 11th International Conference on*, IDC '12, pages 98–105, New York, NY, USA, 2012. ACM.

[21] R. Society. After the reboot: Computing education in UK schools. *Policy Report*, 2017.

[22] A. Sullivan, M. Elkin, and M. U. Bers. KIBO robot demo: engaging young children in programming and engineering. In *Proceedings of the 14th IDC*, pages 418–421, 2015.

[23] D. Wang, T. Wang, and Z. Liu. A Tangible Programming Tool for Children to Cultivate Computational Thinking. *The Scientific World Journal*, 2014, 2014.

[24] D. Wang, C. Zhang, and H. Wang. T-maze: a tangible programming tool for children. In *Proceedings of the 10th international conference on interaction design and children*, pages 127–135, 2011.

[25] D. Wang, L. Zhang, C. Xu, H. Hu, and Y. Qi. A tangible embedded programming system to convey event-handling concept. In *Proceedings of the TEI'16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction*, pages 133–140, 2016.

[26] D. Wang, Y. Zhang, T. Gu, L. He, and H. Wang. E-block: a tangible programming tool for children. In *Adjunct proceedings of the 25th annual ACM symposium on User interface software and technology*, pages 71–72, 2012.

[27] J. Wing. Research notebook: Computational thinking—what and why. *The Link Magazine*, 6, 2011.

[28] J. M. Wing. Computational thinking. *Communications of the ACM*, 49(3):33–35, 2006.